Inter-Organisational Intrusion Detection using Knowledge Grid Technology

Structured abstract

Research Paper

Purpose

This paper introduces a solution for employing Intrusion Detection Technology across organisational boundaries by using knowledge grid technology.

Design / methodology / approach

Employment of Intrusion Detection Technology is currently limited to inside organisation deployments. By setting up communities, maintaining trust relationships between network nodes anywhere in the Internet, security event data, structured into a common XML based format, can be exchanged in a secure and reliable manner.

Findings

A modular architecture has been developed which provides functionality to integrate different audit data generating applications and share knowledge about incidents, vulnerabilities and countermeasure from all over the Internet. A security policy, based on the Chinese Wall Security Policy, ensures the protection of information inserted into the network.

Research limitations / implications

The solution is currently in a preliminary stage, providing the description of the design only. Implementation as well as evaluation is under development.

Practical implications

Trusting communities everywhere in the Internet will be brought into being so that people may establish trust relationships between each other. Participants may decide themselves, whom they trust as a source for security related information rather than depending on centralised approaches.

Originality / value

No approach is known combining the two technologies Intrusion Detection and grid as described in this paper. The decentralised, Peer-To-Peer based grid approach together with the introduction of trust relationships and communities results in a new way of thinking about distributing security audit data.

Abstract. Grid technology has been developed substantially over the recent years and has been pushed further in both scientific and commercial environments. Although the need for handling knowledge within these grid environments has been requested more and more recently, no approach has been published yet, which addresses all problems arising for them. Therefore, we propose a solution for knowledge based grid applications, which attempts to overcome the drawbacks of traditional grid environments when used for knowledge sharing. This paper discusses an idea of a knowledge based grid architecture, which is intended to be used as an infrastructure for exchanging security related information such as intrusion detection audit data, other security related event data and the like.

Keywords: Intrusion Detection, Grid, Security Policy, Trust Relationships, Supplier chains, Peer-To-Peer

Introduction

Intrusion Detection Systems (IDS) (Denning, 1987) have been increasingly employed for security enhancements for a couple of decades already. Besides the unlimited attempts to improve the detection rate by signature based improvements (Lee, Nimbalkar et al., 2000) or aggregation and generalisation (Morakis, Vidalis et al., 2003; Quin and Lee, 2003) on the local side great potential has been noticed in correlating intrusion detection audit data from different locations. Single side intrusion detection systems were more and more substituted by firstly centralised approach and later on hierarchical arrangements of IDS components. (Snapp, Brentano et al., 1991) So called Enterprise Intrusion Detection Systems were brought into being. (Fyodor, 2000; Prelude, 2004) Finally, precisely this enhancement into a hierarchical structure limits the employment to inside organisation deployments. Some decentralised approaches for Intrusion Detection have been introduced distributing very limited functionality of Intrusion Detection capabilities (Snapp, Brentano et al., 1991; Snapp, Brentano et al., 1991; Balasubramaniyan, Garcia-Fernandez et al., 1998). However, the interconnecting of several organisations involved in the exchange of IDS related information is

not yet possible with any current approach although security and trust concerns across organisations have been identified as an issue several years ago already. (Kolluru and Meredith, 2001)

Over the last years hardly any technology in computer sciences has been pushed as much as the novel distributed computing approach grid. The recently performed merging of grid technology with web services with its introduction of so-called stateful web resources resulted in even wider spreading and acceptance of grid systems. The general idea of grid technology is to interconnect several machines to each other in order to solve certain problems as a large unit. Although early developments were mainly limited to scientific problems within very restricted environments such as universities, military institutions and other research undertaking organisations, the grid approach has become more and more popular and is already deployed in commercial and home use environments. People became aware of the waste of unused calculation power on the idle personal computers and started to interconnect them using applications such as S.E.T.I. (SETI, 2005). Basically, the purpose of grid technology is to break down a large problem into small, autonomic pieces, which are processed on several locations within the grid and the results are afterwards merged to one big overall result.

Almost all recent developments in grid computing including decent definitions and descriptions limit the scope of grids to the distribution of computational tasks. It is our belief, however, that the approach of grid may also be applied to the problem of sharing knowledge within grid communities. Consequently, this paper discusses a new approach of a knowledge based grid architecture. In the area of grid technology the Open Grid Service Architecture (OGSA) (Foster, Berry et al., 2004) with its updates towards stateful resources (Foster, Frey et al., 2004) has been commonly accepted as a standard. Taken this as a basis many technologies and procedures could be applied for knowledge grids. The similarities to modern grid technologies are outlined and furthermore the modifications to be made for employment of grid technology for sharing knowledge are presented. Peer-to-peer technology has been used as a network topology for a long time. Peer based networks had been deployed for smaller networks in order to share resources long time before central machines and communities or so called domains have been introduced. Lately, the approach of peer-to-peer computing has grown in popularity due to extensive use of file sharing protocols and applications throughout the Internet. The utilisation of peer-to-peer networking for a grid network

topology provides a very robust and reliable communication base. The definition of the knowledge as the resource for the grid services redefines the way of approaching knowledge based grid environments.

This paper analyses the concept of the Grid for Digital Security (G4DS) – a scalable solution for exchanging security related knowledge such as IDS audit data between organisations – in detail. It defines the interfaces for interaction for communication for both, between the modules of the topology and for using the G4DS with Knowledge Grid applications. Furthermore, it provides information about organisational issues to be solved when establishing such an infrastructure. Before bringing a new Knowledge Grid Service into being, certain questions have to be discussed before any technical measures may be initiated.

Some parts of the idea for the Knowledge Grid architecture are very much into knowledge service discovery and subscription; these issues are beyond the scope of this paper, indeed we assume that the nodes for the services know already about the services to join in.

First of all this paper discusses the background of grid services, the lack of current grid architecture for application on knowledge distribution problems and our basic idea, how to address this drawback. The descriptions of three distinguish deployment scenarios for so-called Inter-Organisational Intrusion Detection Systems present the need of such a solution. Afterwards, the objectives for the Grid for Digital Security will be outlined. Then, the paper will focus on the technical side of the problem and will show how our solution will face the aforementioned objectives. Finally, a conclusion will document what could be achieved with the approach, what are the limitations at the current stage and what is thought to be addressed by future efforts.

Background

In this section we are introducing the basic concepts of inter organisation intrusion detection. After providing information about three intended deployment scenarios the introduction of IOIDS related expressions will provide a common terminology for the remaining paper.

Deployment Scenarios

Inter Organisational Intrusion Detection is thought to be beneficial in very different employment areas. From the interconnection of event loggers from different companies up to the connection of computers of home-users connected to the Internet is considerable. The three following scenarios have been chosen in order to present the flexibility in employment environments for IOIDS:

1. Interconnection of Intrusion Detection Systems (Denning, 1987) for an entire Supplier-Consumer chain.
2. Integration of security related information from academic institutions on the one hand or collections of commercials on the other from all over the Internet for improving detection rates and counter measuring.
3. Open communities for private end user protection.

The following three subsections describe the mentioned scenarios in detail.

Inter Organisational Intrusion Detection for Supplier-Consumer chains

After implementation of Inter Organisational Intrusion Detection near future, a requirement for providing security related information might be part of agreements between several parties, such as convenient for the relationship between suppliers and consumers and finally, the entire chain of supplier parts of a product travel until it ends up at the final consumer. Figure 1 - Supplier Consumer Chain pictures the problem in more detail:

Figure 1 - Supplier Consumer Chain mirrors a variety of real world situations. It might be a chain of manufacturers for "touchable" products such as cars, electronic devices or whatever, it might be companies dealing with and reselling paper values such as banks or insurance companies or even the dealing with knowledge is considerable in today's circumstances where knowledge is very important and comes with much power.

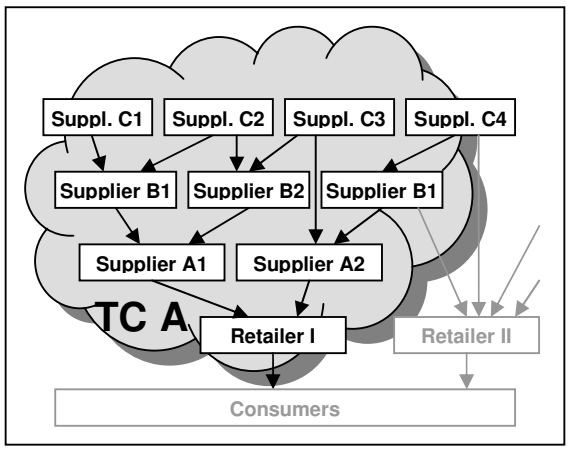

Figure 1 - Supplier Consumer Chain

In the drawn picture *Retailer I* might simply demand for each of its potential suppliers to provide a certain amount of security incident related information about the organisations infrastructure. Moreover, the suppliers of the suppliers are requested to provide the same kind of information. This approach will be pushed to a certain grade and the retailer can finally make sure that it will be informed whenever some kind of significant behaviour has occurred. A policy will be brought into being which the supplier on the one hand but also the retailer on the other have to align to. Finally, a Trusting Community (in the example named *TC A*) will be brought into being.

The following requirements for this kind of infrastructure become obvious already when considering this situation only:

- Each node must be totally confident about the location the data was originated at and based on this knowledge decides about the processing and integration of the data.
- Information from several types of audit data generating applications must be able to be processed and integrated.
- Any supplier or retailer must be enabled to share information with nodes from several Trusting Communities.

Sharing of security related information between academic institutions

Another deployment scenario draws the attention to the problem of missing facilities for exchanging security related information between organisations on a very abstract level. Although organisations such as Computer Incident Response Teams (Cert) (Cert, 2004), Common Vulnerabilities Exposes (CVE) (CVE, 2004) and Information Sharing and Analysis Centres (ISACs) (IT-ISAC, 2005) are supporting this process by providing common names and descriptions

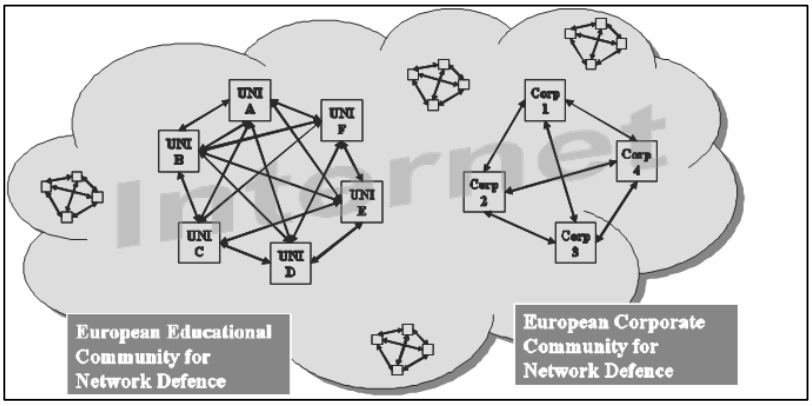

Figure 2 - Examples of Trusting Communities

the actual process of publishing and gathering information is left to humans; hence, it involves a reasonable amount of manual intervention.

Introducing a “European Educational Community for Network Defence” would be able to cut down this amount of time. After agreeing upon a policy for exchanging security related information (including high-level message formats, roles of nodes, trust relationships, etc) and deploying them into a community specific protocol, knowledge will be exchanged automatically.

The same approach is considerable for a reasonable number of companies throughout a certain geographical area such as the European Union. Once they became aware of the potential benefits of exchanging incident, attack and countermeasure information between each other, they start exchanging this type of knowledge and will face the problems of modern threats as a unit rather than isolated from each other.

The two scenarios do not exclude each other. This way, certain universities might also contribute knowledge to the corporate community or vice versa. After all, it will depend on the policies being in place in both communities. Trust within these so-called Trusting Communities must never be undermined by any node.

In fact, the information or knowledge maintained and provided within as well as across the communities belongs to everybody and nobody. It is comparable with the approach of open source software, which has been gaining lots of popularity over the recent years. Everybody may contribute and benefit from the infrastructure and no single node can take it down. The behaviour of a community is based on the policy, which the initial members have to agree upon.

Open communities for private end user protection

Nowadays, also end users become more and more aware of the threats they are facing when connecting their machines to the Internet. Enlightenment and availability of free and open-source software for measures such as anti-virus and personal firewalls have provided an enhancement of protection for home computers. (NSS, 2005) Ease of use and simple configuration are pushing this process.

However, Intrusion Detection Technology has not yet reached a significant penetration in the end user market. A lightweight approach for an IOIDS architecture is able to change this situation significantly. End users are no longer depending on the decisions of companies, instead they can create their own communities and this way exchange security related information among each other. They do not need to trust another party about up-to-date information, but they may decide themselves, which Trusting Community they want to join and which members they want to trust as a reliable source of information.

Terminology

Two major expressions are essential and very distinguish for this paper, namely the Grid for Digital Security (G4DS) and the Inter-Organisational Intrusion Detection System (IOIDS). The former one, Grid for Digital Security, describes all the issues, methodologies and technologies for the subjacent architecture. It is a knowledge based Grid architecture which deals with all issues related to provide and secure the communication channel and provides interfaces for distributing knowledge using this infrastructure.

The Inter-Organisational Intrusion Detection System instead is an application running on top of the Grid for Digital Security. It makes use of the provided architecture and provides an infrastructure for exchanging security related information such as incidents, attack descriptions, information about new attacks in general or related information about countermeasure and the like.

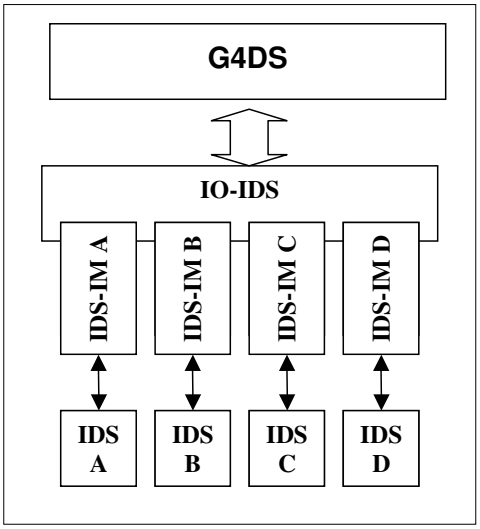

Figure 3 - General Architecture

In fact, the entire system will be made up by the following components (check also Figure 3 - General Architecture):

1. *Grid for Digital Security (G4DS)* – The G4DS represents the fundamental architecture the whole system is built upon. Several issues such as encryption and authorization are addressed in this module. Due to a decentralized approach users of this module will benefit from a robust and reliable architecture. Trust relationships are built up in this module which will enable applications to make publishing decisions based on the role of the members.

2. *Inter Organisational Intrusion Detection System (IOIDS)* – The IOIDS is an implementation that utilises the G4DS. It deals with all issues directly related to Distributed Intrusion Detection Systems such as Intrusion Detection message formats and exchange standards.
3. *IDS-Integration Module (IDS-IM)* – The system will be applicable for a variety of different Intrusion Detection Systems. The integration of the actual IDS is performed by modules which allow an easy plug-in of the different products. At the end of the day, communication can be established between totally different (Enterprise) IDS utilising this Inter Organisations Intrusion Detection System infrastructure.
4. *Connected (Enterprise) Intrusion Detection System (EIDS)* – Currently there are a plenty of Intrusion Detection Systems available implementing different detection and integration technologies. For this research, no separate IDS will be developed but integration of IDS will be performed. Nevertheless the (Enterprise) Intrusion Detection Systems represents one component of the overall solution.

Definitions

In the KGrid environment several roles exist with different permissions. Beside the roles, which may be defined on application basis, there are a few KGrid specific ones for making the system work. The following paragraphs give an introduction to knowledge services and Trusting Communities and then list the various roles with their permissions and responsibilities.

Knowledge Services

A knowledge service is the implementation of one specific application within the KGrid topology. It may span over several Trusting Communities or may include members of certain communities, whereby not the entire TC is part itself.

Knowledge Services define the communication (protocols, algorithms) on a application layer base. This means that the knowledge service itself is described using the Knowledge Service Description Language (KSDL), which is developed as part of this project including the definition of an XML-Schema for KSDL service descriptions. The invocation on this layer, however, is defined using the Web Service Description

Language (WSDL) (Christensen, Curbera et al., 2001). The link between the two documents is maintained by references to the service endpoints in the WSDL document from within the KSDL document.

The Knowledge Service Description maintains information about the following attributes of a knowledge service:

- Unique KGS identifier for the Knowledge Service
- A name for the Knowledge Service
- Current version of the Knowledge Service description
- Current version of the Web Service Description
- Location of the latest Web Service Description
- Optionally, a description
- The knowledge service end points provided by this service and how to invoke them (using which port and services) inside the Web Service Description.
- Information about Service Authorities (SAs) (their identifiers and certificates)
- Information about members and member communities with their identifiers and certificates

Trusting Communities

Trusting communities are an aggregation of nodes, which agree about common purpose and communication protocols. Instead of defining the service properties it defines protocols and algorithms on a lower level of communication. This way, encryption and authentication protocols are agreed upon, which are allowed to be used within the trusting community. Furthermore, agreements about issues such as network protocols are determined here (either to use SOAP, HTTP, SSH, etc.). The responsibility for intercommunity communication including its translation between the protocols is taken by so-called Gateways for Trusting Communities (TCGW).

Each Trusting Community is described in an XML based format with the Community profile. The development of the XML-Schema for this profile is part of the project. The following attributes for a TC are defined in there:

- Unique TC identifier for the Trusting Community

- Name of the Trusting Community
- Current version of the Community Description
- Optionally, a description for the TC
- The Certificate of the Trusting Community containing its public key
- Information about Community Authorities (CAs) (their identifiers and certificates)
- Information about all members of the community (their identifiers and certificates)

Service Authority (SA)

Service authorities have special privileges for maintaining knowledge services (KGS). For each KGS there are at least two SAs. For small and medium-sized communities every node is intended to be a Service Authority. This supports the approach of avoidance of single points of failure. However, policies different from this one may be supported by limiting a certain set of nodes to carry out the responsibility of a SA. Beside tasks performed by each member of a TC SA have to take care of the following additional matters:

- Extension of the lifetime of the KGS. Every service is defined for a certain lifetime (stated in the Knowledge Service Description Document). Each SA is able to extend this attribute.
- Signing new members to the KGS. A new member may request its membership to a knowledge service at any SA of a knowledge service. Once the request is granted, the SA populates the information about the new member of the SA throughout the entirety of nodes subscribed to a service, including its identifier and certificate (with the public key).
- Signing new communities to the KGS. There might be occasions, where it is sensible to add an entire TC to a Knowledge Service rather than single members. This way, any Community Authority of the requesting TC may pass a joining request to any of the SA of the knowledge service. Once the request is granted, the affiliation of the new TC to the KGS is populated throughout the service, including identifier and certificate (with its public key) of the TC.
- Changing roles of members for the KGS. Each member will be equipment with an initial role when joining a Knowledge Grid Service. Depending on the policy for the service this might either be a normal member or a Service Authority. These roles, however, are not static for this member; they might be changed later on. Every Service Authority is able to change the status for a member towards a Service Authority. The other way around, however, may not be performed this way. Since

all Service Authorities have got the same status (there is no hierarchy), once granted Service Authority statuses cannot be revoked from a member later on.

- Changing the KSDL or WSDL, hence introducing new versions of them. During lifetime of a service both its service description as well as it invocation description will change several times. Only Service Authorities are able to introduce and propagate a new version of either of the two descriptions.

Community Authority (CA)

After all, the whole KGrid is peer based, thus central instances are totally avoided. Anyway, for the maintenance of communities the introduction of nodes with special privileges is inalienable. (For a very open community one may still define each member as a CA). CAs are defined at introduction time of a new community. New CAs may be added later on by an existing CA. Each TC must have at least 2 CAs in place. Responsibilities of CAs include but are not limited to:

- Extending lifetime of a Community (TC). As described for the KGS also TC are valid for a certain lifetime only. Each CA is allowed to extend the lifetime of a Trusting Community.
- Signing new members for the TC. Whenever a member is willing to join a Trusting Community it may request membership at any of the CAs for a TC. Once the membership is granted, the information about the new member together with its identifier and the certificate (with its public key) of the member are populated throughout the community.
- Changing roles of members within the TC. Again, new members of a Trusting Community will get assigned an initial role within the TC depending on the policy of the community. However, these roles are not static and may be changed for any member by any of the CAs of a TC. Downgrading the status from a CA to a normal member is not possible since all CAs have the same status and there is no hierarchy in place.
- Changing the description of a TC. The description of a community will evolve during its lifetime. New protocols or encryption mechanisms may be added and these changes have to be mirrored in the description for the community. Changes to the description are supported by introducing a new version of the Community Description. Each CA is allowed to introduce and propagate a new version of the TC Description File.

Members (M)

A member represents any node in the entire KGrid topology. A network node becomes a member as soon as it joins its first community. Members maintain the following attributes:

In general:

- A private key in order to authorise and sign messages.
- The public key to be distributed throughout the different communities and knowledge services.

For each community it is a member of:

- A copy of the community description.
- The certificate for the community issued by any of the CAs of the TC.
- The identities and certificates (with their public keys) of at least 2 CAs of the TC.
- A copy of a list of all members of the TC. This one will change over the time; in fact updates for the member lists will be polled from one of the CAs frequently.
- Its own certificate for this community containing its public key and being signed by any of the CAs of the TC.

For each knowledge service (KS) it is subscribed to:

- A copy of the knowledge service description. (KSDL file)
- A copy of the Web Service Description File for service invocations. (WSDL)
- The certificate for the KS issued by any of the SAs of the KS. (only, if the service is defined as non members inclusive)
- The identities and public keys of at least 2 SAs of the KS.
- A copy of the list of all subscribers for the KS. This one will change over the time; in fact updates for the member lists will be polled from one of the SAs frequently.
- Its own certificate for this knowledge service containing its public key and being signed by any of the SA of the KS.

Trusting Community Gateways (TCGW)

Within the network of communities an additional role for inter community communication is required. The so-called Trusting Community Gateways are responsible for passing messages from one TC to another. The

passing is restricted to certain rules which are defined in the TC policy. A TCGW is always defined for a certain pair of communities and for a certain direction. The rules for passing messages are defined in the TC description itself and are administered by the Community Authorities. The TCGW itself is only an executable role. See section Authentication, authorisation and non-repudiation for messages for more details on Trusting Community Gateways and inter community communication.

Objectives

Knowledge based grids have to address quite similar issues as traditional or computational grids. The following objectives were grouped for our approach for a knowledge based grid:

- Information (knowledge) sharing as the major goal on top of all issues arising.
- Since the exchanged knowledge may consist of very confident or sensitive information, security has to be taken into account as an important issue. This includes but is not limited to encryption, authentication, authorisation, message validation and non repudiation. Research on security for grids has been undertaken by the Open Grid Service Architecture Security Working Group (Nagaratnam, Janson et al., 2002) and especially for communities in “The Anatomy of the Grid – Enabling Scalable Virtual Organisations” (Foster, Kesselman et al., 2001) and “VOMS, an Authorisation System for Virtual Organisations” (Alfieri, Cecchini et al., 2003) and their achievements will also be used for addressing the security goals in knowledge based grid environments.
- Reliability is addressed by “ranging from client-server to peer-to-peer technology” (Foster, Kesselman et al., 2001). Moreover, consistence of the information must be guaranteed throughout the whole grid network.
- It is very important that the members of a community may trust the shared information and that they are furthermore able to assess the trustworthy of information based on the source of information. This can be achieved using trust relationships which were in a similar way already described in (Foster, Kesselman et al., 2001) as “flexible sharing relationships”.
- Easy deployment for a wide range of applications and the ensuring of interoperability and flexibility. This includes (Foster, Kesselman et al., 2001):
    - The concealment of functionality for overlaying implementations in all layers of the approach.

- An approach totally based on standards and the creation of an open architecture for employment with other implementations
- Easy to use interfaces

Furthermore, objectives already described in (Foster, Kesselman et al., 2002) have to be employed, too:

- Error notification and error handling including rollback mechanisms for ensuring consistency of information throughout the entire grid network
- Address naming and address resolution. In this point of view further issues arise when taking into account the requirement of protecting identities as described in more detail in (Pilgermann and Blyth, 2004).
- Upgradeability and compatibility between versions.
- Authorisation and controlling the flow of data.
- Concurrency control, on the one hand within communities, but also, on the other hand, for traffic across community boundaries.
- Scalability, which enables the employment of the approach for both local and large scale networks.

More objectives are often described for modern grid topologies, which are not yet included in our approach due to the differences between traditional and knowledge based grids. These include:

- Quality of Service – since we are not dealing with calculation power or storage sharing, the Quality of Service might only be applied to the whole grid network in the view of availability and the like, but for the knowledge services no QoS is employed at the current stage.
- Since everybody contributes knowledge to the grid and also gathers information from it no accountability is considered to be employed so far. Later developments, however, may come up with some kind of assessments for the contribution and gathering and accountability on top of these values.
- Delegation of authentication credentials are essential in traditional grids for performing processes on behalf of users and, this way, initiate new processes on other locations with these credentials without verifying them each time again. For knowledge based grids, however, this issue is not thought to be of high interest.

Finally, there is the need to prove grid technology as an adequate and useful approach for sharing knowledge across large-scale networks. Therefore, an evaluation and revision process on top of the ongoing research has to be employed, which will assess achieved results and compare them with existing characteristics of traditional grid approaches.

Architecture

Topology

As outlined before, the members are grouped in communities. A community is the group of nodes which all found together because of sharing common interests. Finally, they agreed about the exchange of a certain kind of information and applied some agreements about initial roles of members and how to maintain and develop the community in future time.

Communities consist of usual members without any special privileges (that doesn't touch the privileges on application layer) and the Community Authorities (CA). Each community has at least 2 CAs; responsibilities are described above. There is no upper limitation for the number of Community Authorities; in fact, for small and medium scale topologies each node is supposed to represent a Community Authority, which mirrors the peer based attitude of the approach and makes it very attack resistant against attacks. The changes to be applied for establishing large scale communities (approaches such as the widely employed hierarchical one could be considered) are beyond the scope of this paper.

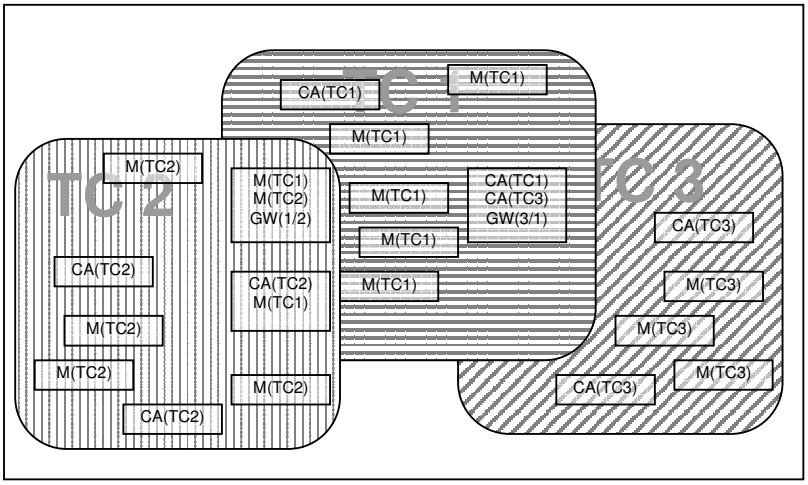

Figure 4 - Overlapping of Trusting Communities

Furthermore, members, no matter which role they are, may belong to several TCs, hence, they establish an interconnection between communities. They are only allowed to forward any messages across community boundaries if they have got the additional role Gateway. (In fact, they must have the Gateway role for this particular pair of communities.) The forwarding of the message must align to the forwarding policy provided by the CAs of the TCs in conjunction with the settings made in the message. Mappings of privileges from one TC to another one are defined in the TCs; however they are cached on the TCGWs. Each message travelling inside the KGrid topology has to be signed by the sender of the message as well as by each gateway it is passing for crossing communities.

Figure 4 - Overlapping of Trusting Communities describes the basic idea of Trusting Communities, the possibilities of memberships of different nodes and finally the overlapping of different communities with its message population across community boundaries.

IOIDS architecture in detail

As outlined in section Terminology already, the IOIDS architecture is strictly built up using a modular approach. This way, whenever a message is sent from one party to another, several parts of the overall architecture are involved. The following section will describe this interaction in 4 steps:

1. Overview of the involved parties and components (includes all components of the overall architecture)
2. Information about the processing of information within the IOIDS layer itself
3. Focus on the components of the subjacent Grid for Digital Security (G4DS) layer involved with the interaction.
4. Finally, some information is provided for interaction with any subjacent (Enterprise) Intrusion Detection System (EIDS); hence, which kind of information has to be put through to the EIDS, what information is gathered from there as well as what special information might raise an action on the overlaying IOIDS layer.

1. Overview of parties and components involved.

Figure 5 - Communication between nodes provides an overview for the communication between two nodes whenever application data between two IOIDS nodes is to be exchanged. Issues such as Trusting Communities are not addressed with this drawing; however, they will be addressed further down in detailed descriptions.

Basically, the exchange of information is comparable with the ISO 7-layer OSI model (Piscitello and Chapin, 1993; Wright and Stevens, 1995). Although, a logical connection is established between the IOIDS components of the two nodes, they are not able to communicate with each other directly. Instead, they are making use of communication facilities provided by the G4DS layer.

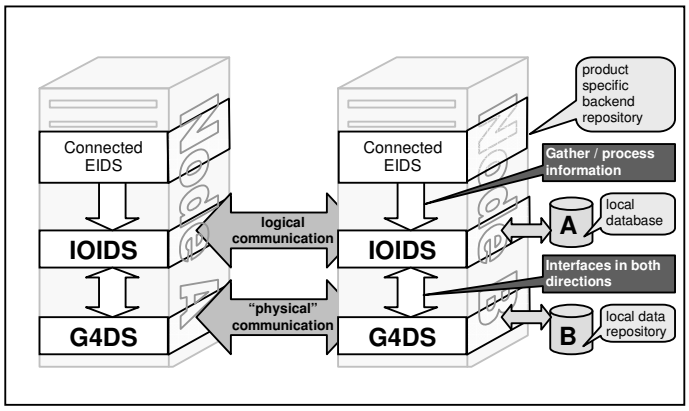

Figure 5 - Communication between nodes

Furthermore, the IOIDS components gather and process information from connected Intrusion Detection Systems. The so-called EIDS components in *Figure 5 - Communication between nodes* may represent a variety of sources of audit data:

- An Enterprise Intrusion Detection System, integrating intrusion related data from different locations all over an organisation
- An Intrusion Detection System employed to monitor a particular, very important node inside an organisation
- Any program undertaking logging of data which is related to security issues.
- Any other tool which is able to be integrated with the IOIDS model; hence, it provides the correct kind of information, the provided information may be transformed into the IOIDS common data format and the corresponding tool is both allowed and supported by the employed policy for the Security Knowledge Service.

Since the integration of subjacent Enterprise Intrusion Detection Systems (or other tools and software) is performed using a modular approach, any useful source of information may contribute knowledge, however, a plug-in has to be implemented for each of them (for more details see section *Integration of knowledge from connected third party event generators*).

2. Communication on IOIDS layer

Basically, each node in the overall topology may act as both a source and a consumer of knowledge. However, regarding to roles as specified in policies for Trusting Communities and Knowledge Services and Sharing of security related information between academic institutions) certain nodes may only be allowed to either send information or receive information.

Each IOIDS node maintains its database for storing all information about incidents, attack descriptions, countermeasures, etc. (marked as '**A'** in *Figure 5 - Communication between nodes*). This database maintains information about both local events as well as events occurring remotely on any node throughout the Knowledge Grid topology. More details about the database architecture are discussed further down in the section IOIDS Database layout.

Figure 6 - Architecture of IOIDS Module visualises the major components of the Inter-Organisational Intrusion Detection System Module. The following section describes all components in brief and provides an insight to responsibilities of each of them:

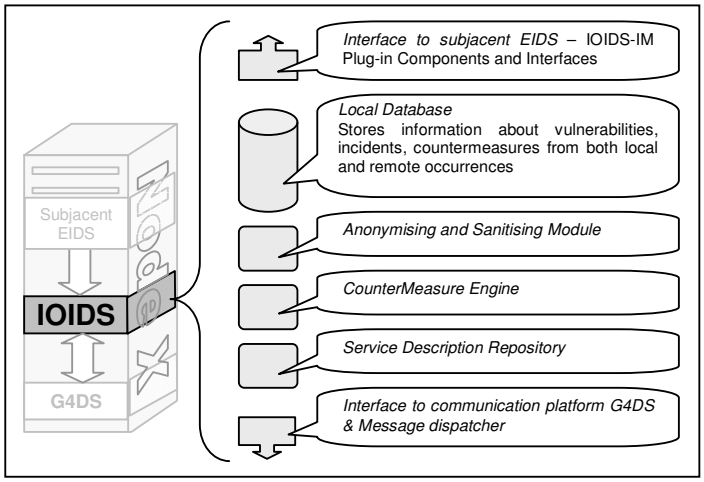

Figure 6 - Architecture of IOIDS Module

- *Interface to subjacent (Enterprise) Intrusion Detection System:*

  The Inter-Organisational Intrusion Detection System architecture is supposed to be connected to a locally deployed Intrusion Detection System. In order to allow the integration of a variety of products from different vendors, a plug-in mechanism was put into place. Interfaces are defined which have to be

implemented for integrating a new product. In order to connect a new product, a so-called IOIDS-Integration Module has to be developed. It is supposed to implement the aforementioned interface and, mainly, care about the transformation between the local IDS specific data format and the common IOIDS data format and of course vice versa. After implementing the IOIDS-IM this product can be registered with IOIDS core and will be handled as part of the overall topology. (for details see also Integration of knowledge from connected further down in this section.)

- *Local Database:*

  The Local Database of the IOIDS core is the central point of storage for this node. In more detail it holds information concerning the following matters:

  - Detected attacks
  - Vulnerabilities in general
  - Information about ways to identify attacks (as for example signatures for rule based Intrusion Detection Systems)
  - Suggestion for countermeasure procedures
  - Relations between these pieces of knowledge

  Basically, the architecture of the database will be aligned to the Intrusion Detection Message Exchange Format (IDMEF) (Curry, Debar et al., 2002) since we are dealing with information about intrusions (and related) and IDMEF is on the one hand able to transport most of the information to be exchanged for our needs, however, on the other hand it allows extensions for the bits which are not addressed directly. In subsection IOIDS Database layout you may find detailed information about the layout and the information to be stored in the IOIDS core database.

- *Anonymising and Sanitising Module:*

  Very sensitive information is to be exchanged using the IOIDS infrastructure. Organisations are forced to take care of their assets, their reputation and trustworthy. This way there will be situations when organisations totally need to unlink information about threats or attacks from their identity; however, in order to protect other node's infrastructure the corresponding information shall be shared. To address this need, this *Anonymising and Sanitising Module* is introduced. Anyway, most of the anonymising is performed in G4DS layer; however, parts of the sanitizing are application specific and need to be addressed on IOIDS layer directly. After all, this topic is beyond the scope of this paper; the module has

been listed for purposes of a complete overview only – finally, the basic idea of anonymising and sanitising was discussed in an earlier publication (Pilgermann and Blyth, 2004).

- *CounterMeasure Engine:*

  The IOIDS system itself will be able to carry out countermeasure actions. For these purposes this separate module has been introduced. The additional value in comparison to other modern countermeasure engines with their support for e.g. TCP session termination or firewall reconfiguration is the opportunity to exchange information across the IOIDS application infrastructure. Having the ability to describe countermeasure actions in a common way and to execute them on many sides in conjunction with the opportunity to interlink them with certain attacks, vulnerabilities or detection descriptions pushes the benefits of countermeasuring to a new level. Countermeasure will be briefly addressed when we talk about message formats for exchanging security related information; however, the architecture and implementation of the countermeasure engine itself is beyond the scope of this paper.

- *Service Description Repository (SDR):*

  The Service Description Repository maintains all these bits of information related to the distribution process throughout the grid architecture, which may not be abstracted down to the G4DS layer; hence, they are application specific. For example, roles for the messages have to be defined and applied to each message to be sent. The following list is a collection of bits of information to be stored in the SDR for the Inter Organisational Intrusion Detection System architecture:

  - Roles of messages and rules for mapping
  - Rules about sanitising
  - Rules and patterns for determination of classifications and destination communities

- *Interface to communication platform G4DS:*

  Finally, the IOIDS subsystem must be connected to the subjacent communication platform G4DS. In order to allow integration with other platforms this issue will be solved using its own module, too. The employment of a dispatcher will perform the processing of incoming message sent over through the Grid for Digital Security.

IOIDS Database layout

The backend for providing persistence for the IDS audit data is implemented by an XML based database. Using the communication mechanism SoapSy (Avourdiadis and Blyth, 2005) data from heterogeneous sources may be processed by this database. The SoapSy approach proposes an architecture consisting of a core, which makes up the static part of the database layout, and several extensions, which are referred to the dynamical part, respectively. In fact, for each different reporting application (so-called agent classes) an extension in the database schema is created and maintained. So far, extensions for syslog, windows event log, snort and the like are considered. The IOIDS data will simply be integrated with this approach using a new IOIDS extension, the so-called IOIDS database sub-schema. It enables the storing of distribution related information which cannot be stored in the SoapSy core. The SoapSy core itself stores information about the source, the destination, the observer and the reporter with classification of each as well as the data and a timestamp for an event. The IOIDS extension, however, is in charge to maintain knowledge of the following type:

- IOIDS network wide unique IOIDS identifier of the message
- Trusting Community for an event
- Classification of an event
- Sender of the event (ID or anonymised ID)
- Is the event anonymised
- Is the message sanitised
- Link to the corresponding sanitised version of the message
- Link to the corresponding unsanitised version of the message

The communication with the SoapSy database is performed using the SOAP (Box, Kakivaya et al., 2000) protocol. A SOAP connection is established to the SOAP handler, which itself passes the information to the database manager. The database manager is in charge to translate the XML formatted request into a database readable format, in fact the Structured Query Language (SQL) (Groff and Weinberg, 1999). The subschema (extensible) to be used is identified by the namespace given in the SOAP request. Regarding to (Avourdiadis and Blyth, 2005) *Figure 7* visualises an example of an event as it would be passed to the

SoapSy Engine. It has to align to the XML schema provided for the IOIDS subschema, which is developed as part of this project. The XML schema for the IOIDS extensible enables the SoapSy engine to configure the database appropriately for the new extension.

```
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV=http://schemas.xmlsoap.org/soap/envelope/
   SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
   <SOAP-ENV:Body>
      <log>
         <sensor>sensor1.comp.glam.ac.uk </sensor>
         <ioids xmlns:ioids="http://www.soapsy.org/ioids" isanonymised=”0”>
            <ioids:date>19/05/2005 16:44:13.234</ioids:date>
            <ioids:messageid>57349085723</ioids:messageid>
            <ioids:classification>2</ioids:classification>
            <ioids:sender>
                  <ioids:memberid>1234567890</ioids:memberid>
                  <ioids:tcid>abcdefg</ioids:tcid>
            </ioids:sender>
            <ioids:sanitising enabled=”0”>
                  <ioids:sanitisedmessage available=”1”>
                        <ioids:messageid>07823475230236</ioids:messageid>
                  </ioids:sanitisedmessage>
                  <ioids:clearmessage available=”0”/>
            </ioids:sanitising>
         </ioids>
      </log>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figure 7 - SOAP request for storing event into SoapSy

Furthermore, it maintains its own additional database with all information not directly related to an event, such as list of Trusting Communities with their members. In *Figure 5 - Communication between nodes* an additional repository has already been introduced marked as *A*.

3. Communication on G4DS layer

The G4DS layer represents an abstraction layer for a secure and reliable communication mechanism independent from any application. The following issues are addressed here:

- Authentication between the communication participants.
- Encryption of all messages.
- Signatures for providing non-repudiation and message integrity.

These issues suggest the employment of a Private Key Infrastructure (PKI). (Rhee, 2003) Although the basic approach for PKI has been employed for the G4DS sub-layer, one significant modification was applied;

namely the abortion of Trusting Authorities. Trusting Authorities are central instances in a network topology and using them would undermine the peer-approach taken.

For the integration of our Trusting Communities together with the Public Key Architecture, the following information is stored on each node:

- Trusting Community Pool holding information about all TCs the specific node is a member of
- For each Trusting Community:
    - ID of the TC
    - Name, version and description of the TC
    - Some information about Creation date, time, initiator and life time and expiring
    - Administrative information such us authorities
    - Information about subscription policy and process
    - A list of the services available in this TC
    - Some status information about updates and current members
    - List of members
- Each item on the member list within the TC description holds the following information:
    - ID of the member
    - Name of the member
    - Public key of the member

Authentication, authorisation and non-repudiation for messages

After introducing the components and interfaces we want to demonstrate the method of operation by passing a message from one node to another one, explaining in detail what is processed in each of the modules involved. Although this section is mainly about details for the G4DS module we also mention the actions undertaken inside the other layers briefly in order to maintain the connection between them.

Consider the following situation: A node of the IOIDS infrastructure (from now on named Source **S**) has connected a variety of different sensors to its central event database. A data mining and data merging engine DME integrated in the IOIDS Knowledge Processor **IKP**, is constantly correlating data from all the

data sources. New messages (events), as results of correlation of other messages, are inserted into the node's central database all the time. At a certain time, the IKP becomes aware of a message, which should be of interest to one (or several) node(s) of the IOIDS infrastructure (The decision about populating knowledge is based on rules, dealing with the knowledge involved, and decent data mining technologies. The method of operation of the DME itself, however, is beyond the scope of this paper).

Before any message may be propagated throughout the network, it has to be protected; hence, the appropriate values for the destination Trusting Community first of all, and the classification of the message afterwards, have to be determined. This action is performed by the Knowledge Protection Engine **KPE**, which bases its decisions on rules. It has to be configured before bringing the IOIDS on the corresponding node into work and certain patterns and information such as address spaces have to be provided. Check the sections *Determine the destination Community of a message* and *Determine the classification of a message* for details on the method of operation of the Knowledge Protection Engine.

Figure 8 - Message travelling between two nodes images the steps for sending a message in detail. It only includes the components of each module involved in the process of sending the message – it does not mirror a complete picture of the overall architecture. Once the message has been equipped with the appropriate values for community and level of protection it is passed to the Message Forwarder and Dispatcher **MFD**. By means of the classification, which is included in the message, the MFD may utilise the Knowledge Sanitising Engine **KSE** for creating a sanitised version of the message. (Again, the detailed description of the KSE is beyond the scope of this paper; however, you may find more detailed information about the process of sanitising in (Pilgermann and Blyth, 2004).) Finally, the message leaves the IOIDS module and is passed from the MFD to the Grid for Digital Security (G4DS) module, performed using a Local Procedure Call (LPC).

For the G4DS module, IOIDS is just one Knowledge Service. It is connected using a well defined interface through the Knowledge Service Integrator **KSI**. Communication between them is realised using Local Procedure Calls (LPC). The descriptions for the Knowledge Service are held in the Knowledge Service Repository **KSR**. For each connected Knowledge Service it holds a Knowledge Service Description

Document (KSDL) and a Web Service Description Document (WSDL). (Also check *Definitions* / *Knowledge Services* for more details on these two documents.)

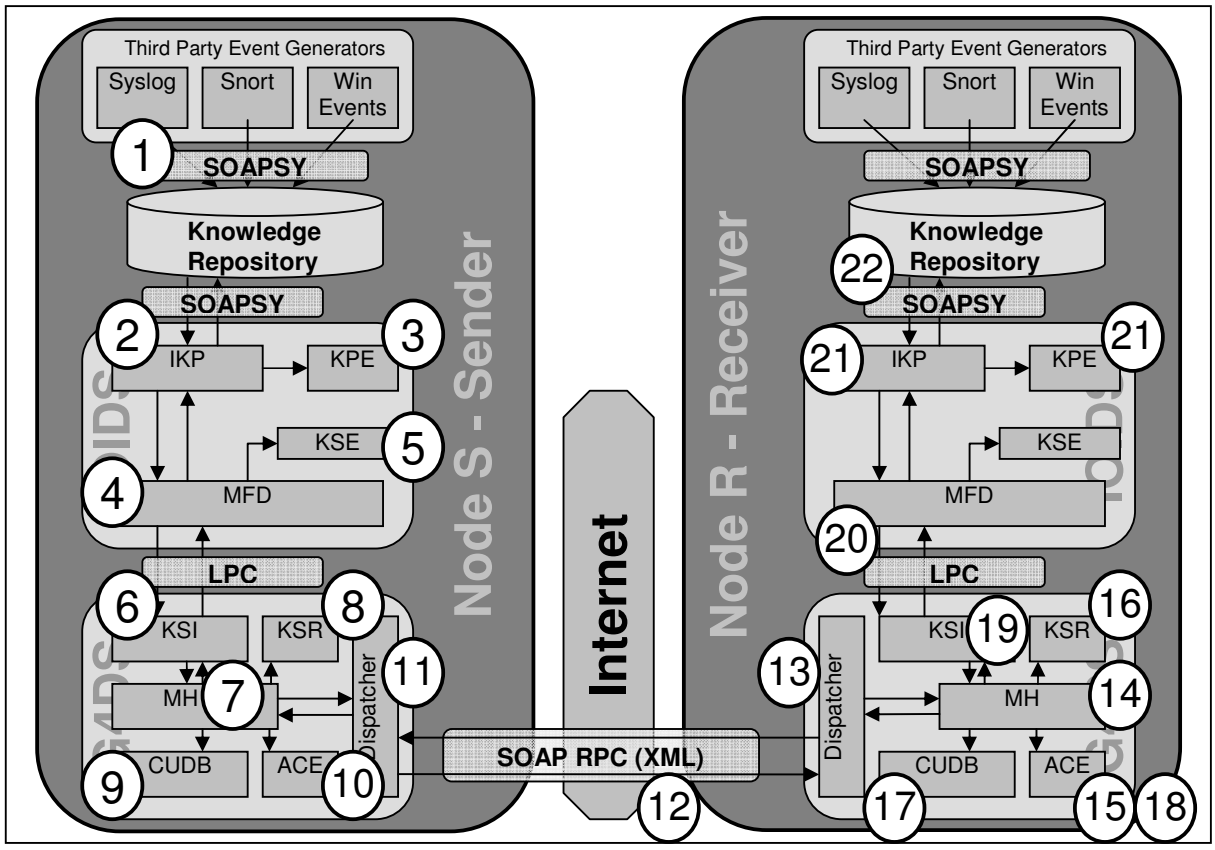

Figure 8 - Message travelling between two nodes

The KSI passes the message to the Message Handler **MH**, which will put together information from the following modules inside the G4DS:

- The Community and User Database **CUDB**: Maintains information about the Trusting Communities (TC) and its members. For each TC there is a Trusting Community Description (TCDL) maintained (More information in *Definitions* / *Trusting Communities*). For each member, the member id, its certificate and its roles are stored. Finally, information about address resolution is provided, which allows the identifier, as used in the connected application, to be resolved into its real address.
- Authentication and Confidentiality Engine **ACE**: Deals with requests concerning authentication of users and confidentiality of messages; hence, tasks such as encryption, decryption, signing and message validation are performed in here.

- Dispatcher: Responsible for passing traffic in both directions, from the network to the Message Handler as well as from the local side to the network. Basically, it is responsible for tasks directly related to the communication such as SOAP server establishment, header and body assembling and the like.

In fact, the Message Handler will first check the application and will gather the corresponding service description documents from the KSR. Afterwards, it requests information from the CUDB about the involved Trusting Community (remember, the KPE of the IOIDS module must have assigned a TC to the message beforehand), and the destination node of the message. By the means of the classification of the message, the keys just gathered will be passed together to the Authentication and Confidentiality Engine (ACE), which is in charge to encrypt and sign the message. Finally, the encrypted and signed message is passed together with the SOAP endpoint information (as gained from the service description documents) of the destination node to the dispatcher, which will establish the connection and pass the message to the receiving node **R**.

The receiver **R** must have performed the same initial steps as S with a central repository running and certain third party event generators logging into it. (The event generators used on R's side, however, may be different from the ones used on S.) Once the IOIDS on R has been started, the G4DS established a SOAP server and has been able to receive messages. We start examining the process step by step exactly at the point, when the Dispatcher (in G4DS) accepts the message from S.

The G4DS Dispatcher accepts the message and unwraps it from the header information whereupon the content is passed to the Message Handler MH. After decrypting the message with its only private key (using the ACE) it extracts the information about the sender's identity and the community used and requests more information about them from the CUDB. As part of it, the certificate (including the public key) of R was returned and the MH passes the message to the Authentication and Confidentiality Engine, for message validation. Afterwards, the message is checked again against the CUDB, to insure that the sender has got the privileges to insert this kind of message. Using the KSR, the correct application for this request may be determined and using the KSI, the message is passed to the connected knowledge service (application) by invoking a Local Procedure Call (LPC).

The IOIDS module on the receiver's side becomes aware of an incoming message in the Message Forwarder and Dispatcher (MFD) module. It passes the message to the IOIDS Knowledge Processor (IKP), which may, based on the rules for the Data Mining procedure, generate a new event for the central data repository and insert the new knowledge into it using the SoapSy interface.

Inter community communication

In the example given before, we simply assumed that the destination node is located within the Trusting Community given as destination community for the message. If this is not the case, however, a few more steps have to be performed. Basically, all the steps on the receiver's side down to the G4DS are kept the same; just the actions within the Message Handler and in the Dispatcher of the G4DS will change as follows.

The G4DS Message Handler always performs a check of the Destination Trusting Community defined in the message, and the Trusting Communities available for the destination node. If the destination node is not a member of the TC as defined in the message, the delivery is rejected and an error is reported to the calling application. If the destination node hits the requirements for communities, the MH checks the affiliation of itself to the destination community. If it is a member of the destination community, then this community will be used to deliver the message directly and the situation as exactly described in *Authentication, authorisation and non-repudiation for messages* is present. If not, however, the message has to be routed through different Trusting Communities to reach its final destination.

The routing in its basics is comparable with the Routing Information Protocol (RIP) (Hedrick, 1988; Stevens, 1994). For each known Trusting Community the Community Authorities (CAs) attempt to find a route through different gateways to reach a node inside the destination community. For these purposes, directly connected TCs are requested to produce some information about the communities, they may reach. (Of course, this assumes, that there are directly connected TCs available, if not however, the TC would act totally isolated and can't reach any other TC anyway.) Once a route has been calculated, the first hop of this route (including the Trusting Community and the gateway(s) TCGW to be used) is put in the Community Description. (In more detail of course some metrics have to be introduced and maintained as well; those details however, are under development and beyond the scope of this paper.)

For the communication on G4DS layer, two basic types of message are distinguished:

- Control messages
- Application messages

Control messages are messages to be processed within the G4DS layer. They will not be passed to any connected application. Application messages, respectively, are attempted to be passed to one of the connected applications.

Whenever a message is to be routed through different communities, it is wrapped into a G4DS control message and marked as to be forwarded. The receiver of the message (in fact, the first hop on the route – namely the TCGW connecting the source community with the first community on the route towards the destination hop) is going to unwrap the message and attempts to deliver the message directly. If not possible, the same procedure is performed again and again; the message is wrapped into a G4DS control message and passed to the next hop on the route towards the final destination. Each hop on the route will gain information about the next hop from the corresponding Trusting Community Description of its community. The approach, in its employment, shows many parallels to the Inter Protocol Routing (Stevens, 1994) as employed for the Internet.

Confidentiality hereby is provided on two layers; namely End-To-End confidentiality and Point-To-Point confidentiality. The actual message to be passed is always encrypted before any process towards sending it is performed. Since the public key of the final destination is utilised for these purposes, only receiver R may decrypt the message and End-To-End confidentiality can be guaranteed. The only information left in plain of the message is the final destination. It is used to pass (or route) the message through the communities. For each hop on this route, the Message Handles of the G4DS module will perform an encryption of the message with the public key of the next hop on the route. This way, only the selected TCGW is able to handle and forward the message.

Further Issues

There are further issues arising when putting up a solution for Inter Organisational Intrusion Detection Systems. For the following issues, ideas have been created; however, they are still under development.

First of all, there will definitively be the need for broadcasting messages to more than one receiver. Especially, the propagation of knowledge about new security breaches is likely to be of interest for a large number of nodes. Broadcasting, however, must be supported throughout different communities. The solution for broadcasting messages as employed for IP networks (Stevens, 1994) is under examination together with ideas for multicasting (Stevens, 1994), which allows a selection of nodes for grouping the receivers.

Furthermore, a unique addressing scheme has to be employed throughout the entire G4DS topology. Each item in the topology (member, community, service (application)) is equipped with an identifier, whereby it is essential to avoid any confusion between them. As part of it, a procedure has to be developed, to resolve certain identifiers into real addresses, such as the identifiers of members into their corresponding IP addresses. The way to approach the resolving bit is, at the current stage, a static list of members in the Trusting Community Description, where for each member either an IP address or a DNS name must be defined.

4. Integration of knowledge from connected third party event generators

Basically, there are two ways to integrate information from local sources such as locally connected Intrusion Detection Systems or other audit data producing applications such as event loggers. The former one is the development of an Inter-Organisational Intrusion Detection System – Integration Module (IOIDS-IM) for plugging into the IOIDS architecture directly. The latter one is the common use of the connected database for the IOIDS event data as well as other applications. Since, the aforementioned SoapSy (Avourdiadis and Blyth, 2005) technology addresses exactly the needs for the latter approach and allows a very easy integration of other audit data with the IOIDS approach only this approach is described here. The former approach has not yet been developed in detail and its specification is beyond the scope of this paper.

As described in (Avourdiadis and Blyth, 2005) many different audit data generating applications may log into the XML based SoapSy database. The advantages of using this approach are:

- The audit data is available in a simply readable and well-defined format.
- The audit data may be accessed very easily through an XML based SOAP interface.
- Its implementation is unlinked from the IOIDS architecture; hence, wider employment is likely and due to reuse with other projects, IOIDS will be able to access a variety of different audit data generating resources more quickly.

The major issue arising for using all this information for the IOIDS architecture is grounded in the classification of information and the deriving of IOIDS messages. Rules have to be developed giving the IOIDS subsystem efficient information about how to use audit data, how to create and derive messages from both, local and remote resources, how to assign the new data to a Trusting Community and, finally, how to protect the new piece of information adequately. Further issues arise when drawing attention to anonymising and sanitising objectives, which require a further breaking down of protection policies to come up with more detailed rules for assigning a classification for messages.

In detail, the following objectives have to be addressed when integrating knowledge from other data sources:

- The IOIDS subsystem must take notice of modifications or additions made to the database.
- The IOIDS subsystem must be able to extract the information from the core for each message, but also should be able to access information from extensibles of other subsystem in order to gain additional, more detailed information about an event.
- Information from the core (and also from the other subsystems) has to be transformed in the IOIDS subsystem and additional information has to be assigned for the IOIDS subsystem in order to allow appropriate processing of the messages later on.
- The IOIDS subsystem must take care of protecting the local knowledge efficiently; hence no knowledge must leave the local database and being propagated throughout the IOIDS network before it has been approved by the knowledge assessment engine of the IOIDS infrastructure.

The protection of local knowledge is a major issue arising. The following measures are applied in order to cope with this issue:

- No other event then events of the IOIDS subsystem maybe propagated throughout the IOIDS network; hence, each message must first be processed and assessed and an event for the IOIDS subsystem has to be created before it may be send through the IOIDS infrastructure.
- Requests from other notes of the IOIDS network will only be performed on knowledge marked as an IOIDS event. Again, whenever further knowledge is required, the Event Assessment and Classification Engine has to process required events before.
- The propagation of knowledge is strictly limited by the restrictive rules for classification determination. For example, whenever an event is classified as private it well never pass the boundaries of the local node. Further details about destination community and classification determination are provided in the section Security Policy in Detail and Enforcement of its deployment.

The assessment and classification is performed based on rules involving the following attributes of each event:

- Which subsystem was / subsystems were involved for the event. (Windows Event Logger, Syslog, IOIDS subsystem)
- Which addresses or address spaces (IP addresses or DNS names or domains) are involved for the source, destination, agent and manager for an event?
- Which services or ports have been involved?
- Any information about users and operating systems.
- Was the event occurring within a certain time frame?
- Are some predefined addresses or other patterns occurring in the description for an event?

Before a node is brought up running, rules have to be defined and applied, which will give information about how to classify messages and, furthermore, how to assign them to certain Trusting Communities. The first five items in the list above point directly to a specific kind of information, whereby the last one, using the patterns, enables the user at configuration time to address all kinds of information since a full text search over the entire event description is performed there. Finally, at configuration time a default classification as well as a default destination Trusting Community have to be defined, which will be assigned to the event whenever none of the aforementioned rules apply to the event.

Security Policy in Detail and Enforcement of its deployment

In order to enforce proper distribution, processing and storage of knowledge, the concept of Chinese Wall Security Policy (Brewer and Nash, 1989) has been employed for the approach. The different parties might be either Trusting Communities or single nodes. It must be made sure that information from one side does never cross to another side, even knowledge resulting from processing several messages (unless, the message is marked to public; hence, it is supposed to be sent across TC boundaries).

Organising knowledge

Basically, there are two ways to assign classification and destination Trusting Community to a new message; either regarding to all pieces of source information a set of privileges is created for each TC involved and a classification tags is set for each individually or the most appropriate destination community is attempted to be identified and only one set of privileges needs to be determined. Using the latter way the most sensible Trusting Community is chosen to be destination TC of the new message and adequate classifications are calculated and kept for this one only. In order to keep the entire system as simple as possible and not to blow up the status information for messages too much, the latter way is employed; hence, each event belongs to a single Trusting Community only.

Each piece of information is tagged with the following information when stored on a node:

- Origin of the message (includes both, the actual sending node of the message and the Trusting Community it was created in)
- A classification for the message itself, which provides information about the ways this message is supposed to be used.

The origin may be determined easily by processing header information of the message which has been sent before. (However, a sending node might be member of several TCs, this way it is required to define into which TC the current message is sent.) The classification for the message must be defined by the sending node. (For more details about available classifications for messages also check Employed Policy and Roles for Security Knowledge Service.) Consequently, the following formalism is introduced:

$$\mathbf{k = \{m, s, t, c\}}$$

These symbols represent the following information; a piece of Knowledge (k) is made up by the combination of:

- The message itself (m)
- A source for this message (s)
- A Trusting Community of this message (t)
- A classification for the message (c) defining, how to handle, process and distribute it

If a new created knowledge chunk k_x is considered, all its related information is represented the following way:

$$\mathbf{K_x = \{M_x, S_x, T_x, C_x\} \rightarrow k_x}$$ for $M_x = \{m_1, m_2, \ldots, m_a)$ with a as number of messages,
$S_x = \{s_1, s_2, \ldots, s_b)$ with b as number of sources,
$T_x = \{t_1, t_2, \ldots, t_c)$ with c as number of TCs involved and
$C_x = \{c_1, c_2, \ldots, c_d)$ with d as number of classifications.

Simplified, *Figure 9 - Knowledge Management* visualises the model employed for managing knowledge within the IOIDS architecture. It does not mirror the entire complexity of the model since the relations between the knowledge chunks have not been provided with the figure.

A knowledge pool (K) is made up by the sets of messages (M), Sources (S), Trusting Communities (T) and their classifications (C). Pieces of information in the knowledge pool have relations between each other. In fact, no item in the knowledge pool may exist in multiple instances; hence, each message is put into relation with its corresponding sender, source Trusting Community and Classification.

Whenever a new message is being created, a certain subset of pieces of knowledge is involved. The overall subset is named K_x with all its members (or sub-subsets) M_x, S_x, T_x and C_x for the messages, sources, trusting communities and classifications involved for creating this new message. In an example as described

in Figure 9 - Knowledge Management 3 source messages (M_x = {M_3, M_5, M_7}) with their 3 source addresses (S_x = {S_2, S_5, S_4}) from two different communities (T_x = {T_2, T_4}) and classifications (C_x = {C_1, C_3}) are involved. Finally, the entirety of the knowledge subset K_x results in the new piece of knowledge k_x. The new piece of knowledge k_x comes as a unit of message, source node, destination community and classification; hence, at the same time new entries will be asserted into the data repository. (At least one new entry for the message has to be inserted; the values for classification, Trusting Community and Source Node may be existent in the database already.)

By storing the information about the source, the trusting community and the classification of a message together with the message itself, it can be made sure that knowledge never escapes from its intended distribution domain.

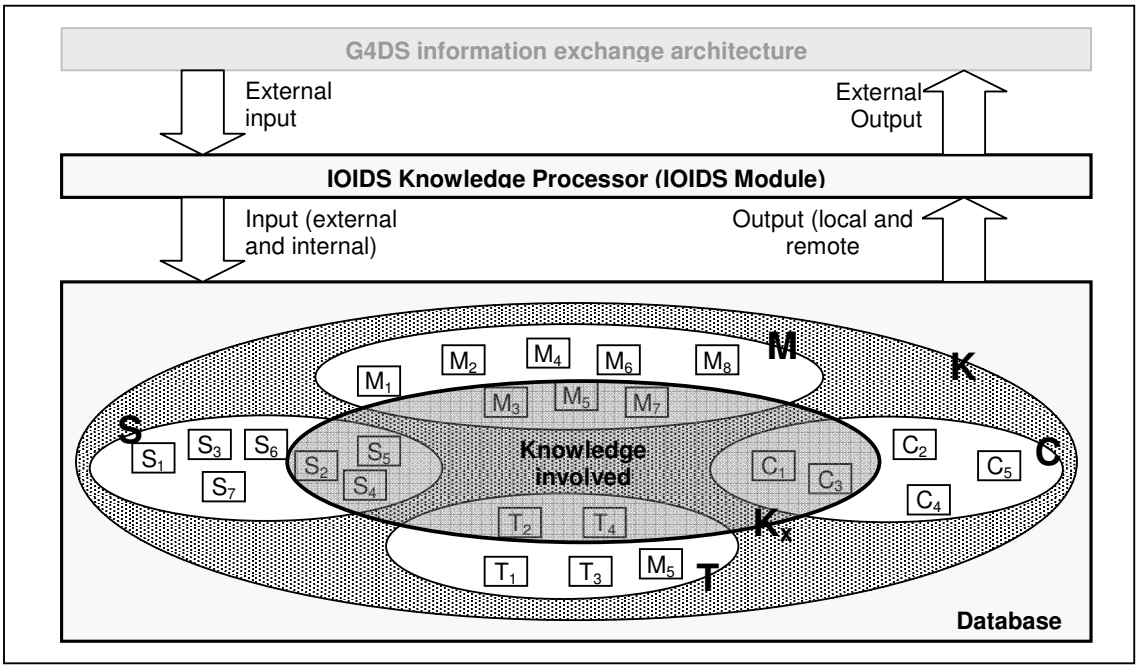

Figure 9 - Knowledge Management

Employed Policy and Roles for Security Knowledge Service

Different classifications provide different levels of protection for messages. The classifications for messages are ordered in a way that a higher number represents a weaker grade of protection. The following protection grades are employed:

Grade	Description
0	Local Confidential – this message never leaves the local node. Nor does any message derived from this piece of information leave this node.
1	Local Confidential, but Sanitised for Third Party – local message needs to be protected completely, however, sanitised message may be sent to destination party.
2	Local Confidential, but Sanitised for Community
3	Local Confidential, but Sanitised for Everybody
4	Destination Confidential – information from this message may only be used and processed on the receiver's node itself. Neither the message nor any message which is (partially) derived from this message may be sent to any other party.
5	Destination Confidential, but Sanitised to Community
6	Destination Confidential, but Sanitised to Everybody
7	Community Boundary Protected – this message may be circulated throughout the source Trusting Community. Any derived message may be handled the same way; however, neither the message itself nor any (partially) derived message may ever leave the community boundaries.
8	Community Boundary, but Sanitised to Everybody
9	-- unused --
10	Public – Practically, no protection. This message may be used, processed and passed all ways. This message will not lead into any restrictions for any message derived from this message.

Sanitising may only be performed by the origin of the message itself. No other node is able or allowed to perform sanitising on behalf of the source node. Even, if no sanitised version if available (or not yet available), but sanitised versions are the only ones to be forwarded, the actual (un-changed) message must not be used to create the derived message.

Determine the destination Community of a message

Before the classification of a message can be calculated its destination Trusting Community has to be appointed. Goal is the highest possible propagation of the knowledge without violating any restrictions made up by the classification rules of all the messages in the knowledge pool K_x for the new message.

Source Trusting Communities as well as classifications of input messages have to be examined. In fact, the value for the destination Trusting Community is calculated by the following function:

$$\{M_x, C_x, T_x\} \rightarrow t_x$$

(Sanitising bits of the classification rules do not need to be taken into account for this procedure since it is assumed that the Trusting Community of a message would be garbled in the process of sanitising). The following rules are applied to the knowledge pool in exactly the given order in order to determine the destination Trusting Community of the new knowledge chunk k_x. (For the formulas, the symbolism as described in (Online, 2005) is used.)

1. There is at least one message in the knowledge pool whose classification is specified as C0 – Local confidential (including its derived grades for sanitising C1, C2 and C3). – The new message will not have any destination Trusting Community. (This message (or any derived message) must not be send to any party; hence, the applying of a Trusting Community is superfluous.)

   $\$ x \hat{I} C_x \mid x \hat{I} \{C0, C1, C2, C3\} \quad \text{Þ } t_x = n.a.$

2. If there are at least two messages with classification C4 - Destination Confidential (or one of its derived grades for sanitising C5 and C6) being originated in two different Trusting Communities, then there won't be any destination TC assigned for k_x.

   $\$ (c_1, c_2, t_1, t_2, m_1, m_2) \acute{I} (C_X, T_X, M_X) \mid c_1, c_2 \hat{I} \{C4, C5, C6\} \& t_1 \rightarrow m_1 \& m_1 \rightarrow c_1 \& t_2 \rightarrow m_2 \& m_2 \rightarrow c_2 \& t1 \neq t2 \quad \text{Þ } t_x = n.a.$

3. There are at least two messages with classification C7 or C8 – Community Boundary Protected whose source TCs are different. – No destination TC will be assigned.

$ (c_1, c_2, t_1, t_2, m_1, m_2) Í (C_X, T_X, M_X) | c_1, c_2 Î {C7, C8} & t_1 → m_1 & m_1 → c_1 & t_2 → m_2 & m_2 → c_2 & t1 ¹ t2 Þ t_x = n.a.

4. There is at least one pair of messages with classification C4 (or C5, C6) for the first one and classification C7 (or C8) for the second one, which are originated in different Trusting Communities. – No destination TC may be applied.

   $ (c_1, c_2, t_1, t_2, m_1, m_2) Í (C_X, T_X, M_X) | c_1 Î {C4, C5, C6} & c_2 Î {C7, C8} & t_1 → m_1 & m_1 → c_1 & t_2 → m_2 & m_2 → c_2 & t1 ¹ t2 Þ t_x = n.a.

5. All messages in the knowledge pool are originated in the same Trusting Community t_1. – The destination Trusting Community equals the one of the source messages.

   t = α " t Î T_x Þ t_x = α

6. Messages from different Trusting Communities are involved; however, only exactly one message is marked with classification C4 – Destination Confidential (or one of its derived grades C5 and C6). – The destination Trusting Community is the one of this specific message.

   $ (c, t_1, t_2) Î (C_x, T_x) | c Î {C4, C5, C6} & t_1 → m & m → c & t_1 ¹ t_2 Þ t_x = t_1

7. All messages with protection C7 / C8 – Community Boundary are originated in the same TC.

   t = α " (t, c) Î (T_x, C_x) | t → m & m → c & c Î {C7, C8} Þ t_x = α

8. All messages are public. – The destination TC is the one with the most entries for processed messages with this Trusting Community.

   c = C10 " (c) Î (C_x) Þ t_x = t " t = Max(m_x, t_x) with m_x → t_x

Determine the classification of a message

In order to determine the classification for a new piece of knowledge two situations have to be considered:

1. A single message has been processed or several messages have been processed in order to create the piece of knowledge; but all messages have go the same classification and are originated in the same Trusting Community.

2. Several messages have been processed in order to create the piece of knowledge, processed messages come from different sources (nodes or communities) and / or have different classifications applied.

The former case is easy to handle; however, the latter one requires more efforts in order to align to distribution policies. Classifications for messages are ordered, a classification with a higher number represents a lower protection than one with a low number; hence, the classification 0 stands for the highest protection of the message ever – means, only the creator itself is using and processing the data. (See Employed Policy and Roles for Security Knowledge Service for details.)

First of all it has to be made clear that all this information is about the privileges of the message and how it may be published to other nodes of communities. The node itself may process all information available on this local node; however, it will be restricted in passing discovered knowledge to other nodes of the communities depending on the sources and classifications of chunks of processed knowledge. The following rules are applied to determine a classification for a new message regarding to the sources and classifications of all messages being used for creating this new message (the rules in here are ordered, this way the first rule applicable for the knowledge pool of the new message K_x will be used and the processing is terminated).

The classification determination is performed by two progressive stages. In the first stage sanitising options will be left behind and only the major class (c_{tmp}) will be calculated. Four major classes are available; namely $C_{tmp}A$ – Local Confidential, $C_{tmp}B$ – Destination Confidential, $C_{tmp}C$ – Community Protected and $C_{tmp}D$ – Public. The second step takes into account all the sanitising information and will this way calculate the exact destination classification. The two functions used for calculating the classification are the following ones:

$$\{M_x, C_x, T_x\} \rightarrow c_{tmp}$$

$$\{c_{tmp}, M_x, C_x\} \rightarrow c_x$$

The function for calculating c_{tmp} is represented by the following rules. They have to be applied in the given order:

1. At least one message of the knowledge pool is classified Local confidential C0 or Local confidential with any of the Sanitising options (C1, C2 and C3). – The temporary classification of the new message is $C_{tmp}A$ – Local confidential.

   $ x Î C_x | x Î {C0, C1, C2, C3} **Þ $c_{tmp} = C_{tmp}A$**

2. At least one message of the knowledge pool is classified Destination Confidential C4 or Destination Confidential with any of the Sanitising options (C5 and C6). – The temporary classification of the new message is $C_{tmp}A$ – Local confidential.

   $ x Î C_x | x Î {C4, C5, C6} **Þ $c_{tmp} = C_{tmp}A$**

3. All messages in knowledge pool which are classified as C7 – Community Protected or its Sanitised classification C8 are originated in the same Trusting Community as the Destination Community of the message. – The temporary classification of the new message is $C_{tmp}C$ – Community Protected.

   $t = t_x$ " (t, c) Î (T_x,C_x) | t → m & m → c & c Î {C7, C8} **Þ $c_{tmp} = C_{tmp}C$**

4. There is at least one message classified as C7 Community Protected or its sanitised classification C8 which is not originated in the destination community of the message. – The temporary classification of the new message is $C_{tmp}A$ – Local confidential.

   $ (t, c) Î (C_x, T_x) | c Î {C7, C8} & t ¹ t_x **Þ $c_{tmp} = C_{tmp}A$**

5. All messages in K_x are classified *public* C10. – The temporary classification of the new message is public $C_{tmp}D$.

   x = C10 " x Î C_x **Þ $c_{tmp} = C_{tmp}D$**

After determining the major destination class, the final destination classification is calculated by applying the following rules in the given order:

1. The temporary classification is Local Confidential $C_{tmp}A$. There is at least one source message with classification C0 – Local Confidential. – The destination classification is C0 – Local Confidential.

   $ c Î C_x | c = C0 & $c_{tmp} = C_{tmp}A$ **Þ c_X = C0**

2. The temporary classification is Local Confidential $C_{tmp}A$. There is at least one source message with classification C1 – Local Confidential but Sanitised for Third Party or C4 – Destination Confidential. – The destination classification is C1 – Local Confidential but Sanitised for Third Party.

   $ c Î C_x | c Î {C1, C4} & $c_{tmp} = C_{tmp}A$ Þ $c_X = C1$

3. The temporary classification is Local Confidential $C_{tmp}A$. There is at least one source message with classification C2 – Local Confidential but Sanitised for Community or with classification C5 - Destination Confidential but Sanitised for Community or with classification C7 – Community Boundary protected. – The destination classification is C2 – Local Confidential but Sanitised for Community.

   $ c Î C_x | c Î {C2, C5, C7} & $c_{tmp} = C_{tmp}A$ Þ $c_X = C2$

4. The temporary classification is Local Confidential $C_{tmp}A$. – The destination classification is C3 – Local Confidential but Sanitised to everybody.

   $c_{tmp} = C_{tmp}A$ Þ $c_X = C3$

5. The temporary classification is Destination Confidential $C_{tmp}B$. There is at least one source message with classification C4 – Destination Confidential. – The destination classification is C4 – Destination Confidential.

   $ c Î C_x | c = C4 & $c_{tmp} = C_{tmp}B$ Þ $c_X = C4$

6. The temporary classification is Local Confidential $C_{tmp}B$. There is at least one source message with classification C5 – Destination Confidential but Sanitised for Community or C7 – Community Boundary Protected. – The destination classification is C5 – Destination Confidential but Sanitised for Community.

   $ c Î C_x | c Î {C5, C7} & $c_{tmp} = C_{tmp}B$ Þ $c_X = C5$

7. The temporary classification is Destination Confidential $C_{tmp}B$. – The destination classification is C6 – Destination Confidential but Sanitised to everybody.

   $c_{tmp} = C_{tmp}B$ Þ $c_X = C6$

8. The temporary classification is Community Protected $C_{tmp}C$. There is at least one source message with classification C7 – Community Boundary Protected. – The destination classification is C7 – Community Boundary Protected.

   $ c Î C_x | c = C7 & $c_{tmp} = C_{tmp}C$ Þ $c_X = C7$

9. The temporary classification is Community Protected $C_{tmp}C$. – The destination classification is C8 – Community Boundary Protected but Sanitised to everybody.

   $\mathbf{c_{tmp} = C_{tmp}C}$ **Þ** $\mathbf{c_X = C8}$

10. The temporary classification is Public $C_{tmp}D$. – The destination classification is C10 – Public.

    $\mathbf{c_{tmp} = C_{tmp}D}$ **Þ** $\mathbf{c_X = C10}$

Destination confidential messages are not created during this process; however, this classification may be assigned whenever a reply is sent to any node in reaction to a knowledge enquiry or it is assigned resulting from a rule for integration of audit data from third party sources (see Integration of knowledge from connected for details). Another occasion is considered with the existence of some piece of information, which is only related to a certain node and it's essential to inform this node; however, this node shall not be enabled to pass on this information nor any message derived from it, not even within the community. The classification "Destination Confidential" provides an opportunity to the application to publish information exactly this way.

Conclusion

Inter Organisational Intrusion Detection is thought to provide great enhancements for securing modern networks and it is our strong belief that the solution, as proposed in this paper, builds up a secure and reliable knowledge exchange platform, suitable to share this kind of very sensitive knowledge over the Internet. Three employment scenarios have been pictured; each of them showing the potential and benefits of creating IOIDS networks throughout the Internet in its very individual view.

The peer-to-peer based communication platform Grid for Digital Security provides an abstract and very reliable communication platform, employable for a wide range of applications. By utilising a modified Public Key Infrastructure its features for authentication, encryption and message validation ensure secure communication including all features provided by PKI such as confidentiality, message integrity and non-repudiation. A high level description of an interface has been provided, which introduces the so-called Knowledge Services, basically, the applications running on top of G4DS.

Much attention has been drawn on the trust relationships between the knowledge sharing parties and Trusting Communities have been introduced. Their method of operation has been discussed in very detail, and by introducing the employed security policy with its measures for the determination of destination community and classifications for new chunks of knowledge the mechanism for protecting knowledge was made clear.

Finally, the overall architecture for the employment of an Inter Organisation Intrusion Detection System running on top of the G4DS was proposed. The modules involved have been distinguished and using a real-world scenario of propagating the occurrence of an event their collaboration has been outlined. There was an insight given into the cooperation of IOIDS with third party event generators and the description of the connectivity through the SoapSy interface provided details about the gaining from as well as contributing to the central event data repository. Measures have been pointed out to assign a classification and community to these new pieces of knowledge using a rule based approach.

An implementation for this solution is ongoing work and a steady evaluation and examination process along the development of the architecture shall prove the applicability of the approach to conquest the security issues and problems faced by everybody using the Internet these days.

References

Alfieri, R., R. Cecchini, et al. (2003). VOMS: an Authorization System for Virtual Organizations. 1st European Across Grids Conference, Santiago de Compostela.

Avourdiadis, N. and A. Blyth (2005). "SoapSy - Unifying Security Data from Various Heterogeneous Distribute Systems into a Single Database Architecture." publication pending.

Balasubramaniyan, J. S., J. O. Garcia-Fernandez, et al. (1998). "An Architecture for Intrusion Detection using Autonomous Agents." ACSAC.

Box, D., G. Kakivaya, et al. (2000). SOAP: Simple Object Access Protocol.

Brewer, D. F. C. and M. J. Nash (1989). The Chinese Wall Security Policy. IEEE Symposium On Research In Security And Privacy, OAKLAND, CALIFORNIA.

Cert (2004). Cert / Coordination Center (http://www.cert.org/).

Christensen, E., F. Curbera, et al. (2001). Web Services Description Language (WSDL) 1.1 W3C, Note 15.

Curry, D., H. Debar, et al. (2002). Intrusion Detection Message Exchange Format Data Model and Extensible Markup Language (XML) Document Type Definition, IETF. 2003.

CVE (2004). Common Vulnerabilities and Exposes (http://www.cve.mitre.org).

Denning, D. (1987). "An Intrusion Detection Model." IEEE Trans. on Software Engineering SE-13(2).

Foster, I., D. Berry, et al. (2004). The Open Grid Services Architecture, Version 1.0. 2004.

Foster, I., J. Frey, et al. (2004). Modeling Stateful Resources with Web Services.

Foster, I., C. Kesselman, et al. (2002). The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. 2004.

Foster, I., C. Kesselman, et al. (2001). The Anatomy of the Grid: Enabling Scalable Virtual Organizations. 2004.
Fyodor, Y. (2000). "Snortnet" - A distributed Intrusion Detection System. 2004.
Groff, J. R. and P. N. Weinberg (1999). SQL: The Complete Reference, McGraw-Hill.
Hedrick, C. L. (1988). RFC 1058: Routing Information Protocol. 2005.
IT-ISAC (2005). IT Information Sharing and Analysis Center (IT-ISAC). 2005.
Kolluru, R. and P. H. Meredith (2001). "Security and trust management in supply chains." Information Management & Computer Security 9(5): 233 -- 236.
Lee, W., R. A. Nimbalkar, et al. (2000). A Data Mining and CIDF Based Approach for Detecting Novel and Distributed Intrusions. Recent Advantages in Intrusion Detection, Toulouse, France, Department of Computer Science (North Caroline State University), Department of Computer Science (Columbia University).
Morakis, E., S. Vidalis, et al. (2003). A Framework for Representing and Analysing Cyber Attacks Using Object Oriented Hierarchy Trees. The 2nd European Conference On Information Warfare And Security (ECIW), Reading, UK.
Nagaratnam, N., P. Janson, et al. (2002). The Security Architecture for Open Grid Services. 2004.
NSS (2005). Internet Security - "The Modern Day Gold Rush", NSS Group. 2005.
Online, M. (2005). Mathematische Symbole und Abkuerzungen (Mathematical Symbols and abbreviations). 2005.
Pilgermann, M. and A. Blyth (2004). Anonymizing Data in a Peer-To-Peer based Distributed Intrusion Detection System - A possible Approach. European Conference on Information Warfare (ECIW) 2004, London.
Piscitello, D. M. and A. L. Chapin (1993). Open Systems Networking: TCP/IP and OSI, Addison-Wesley Longman Publishing Co., Inc.
Prelude (2004). Prelude: an Open Source, Hybrid Intrusion Detection System. 2004.
Quin, X. and W. Lee (2003). Statistical Causality Analysis of INFOSEC Alert Data. Recent Advances in Intrusion Detection (RAID) 2003, Pittsburgh, PA, USA.
Rhee, M. Y. (2003). Internet Security - Cryptographic principles, algorithms and protocols. Chichester, West Sussex, England, John Wiley & Sons Ltd, The Atrium.
SETI (2005). Seti at home. 2005.
Snapp, S., J. Brentano, et al. (1991). A System for Distributed Intrusion Detection. COMPCON Spring '91, San Francisco.
Snapp, S. R., J. Brentano, et al. (1991). DIDS (Distributed Intrusion Detection System) Motivation, Architecture, and An Early Prototype. 14th National Computer Security Conference, Washington, D.C.
Stevens, W. R. (1994). TCP/IP Illustrated, Volume 1: The Protocols, Addison-Wesley.
Wright, G. R. and W. R. Stevens (1995). TCP/IP Illustrated, Volume 2 - The Implementation, Addison-Wesley Publishing Company.

The authors:

Michael Pilgermann, Andrew Blyth, Stilianos Vidalis

Michael Pilgermann has been working as a Research Student in the Information Security Research Group of the University of Glamorgan – School of Computing since 2003 focussing on intrusion detection technologies and distributed network topologies. His research aims to result in a solution for a scalable approach for exchanging security audit data between parties across organisational boundaries.

School of Computing, University of Glamorgan, Pontypridd, CF37 1DL, UK
Email: mpilgerm@glam.ac.uk
Tel No. +44 1443 654086
Fax No. +44 1443 482715

Dr. Andrew Blyth is Head of the Information Security Research Group at the School of Computing, University of Glamorgan, UK. He received his Ph.D. in 1995 from the Computing Laboratory at the Newcastle University, UK. He has published numerous papers in the area of Network Security and Intrusion Detection Systems. His research interests include the capture, analysis and visualisation of security/network related information.

School of Computing, University of Glamorgan, Pontypridd, CF37 1DL, UK
Email: ajcblyth@glam.ac.uk
Tel No. +44 1443 482245
Fax No. +44 1443 482715

Dr. Stilianos Vidalis received his PhD in Threat Assessment in July 2004 from the University of Glamorgan. He joined the School of Computing in 2001 where he is currently employed as a Research Fellow. Dr. Vidalis is a member of the ISRT, and has experience on European R&D projects. Dr. Vidalis is lecturing on the subjects of information security and computer networks in both undergraduate and postgraduate levels. His research interests include information security, threat assessment, network security, effective computer defence mechanisms and intrusion detection systems.

School of Computing, University of Glamorgan, Pontypridd, CF37 1DL, UK
Email: svidalis@glam.ac.uk
Tel No. +44 1443 482731
Fax No. +44 1443 482715